Y-MP Floating Point and Cholesky Factorization
RND-90-007

Russell Carter
Computer Sciences Corporation
NASA Ames Research Center
Moffett Field, CA 94035, USA

Introduction

The Cray 2 and Cray Y-MP are in many respects very similar computer systems, particularly when compared with other computer systems currently available. This similarity extends to the area of floating point arithmetic hardware implementation. The hardware implementations differ only very slightly, but this difference is sufficient to cause certain algorithms to exhibit significant increases in error when run on the Cray Y-MP. This paper describes the differences between the two arithmetic implementations, details the error behavior differences between the two arithmetics, and presents a model of how the floating point error behavior influences the error behavior of the solution of large positive definite systems by Cholesky factorization.

An interesting example was produced by a finite element model of the aeroelastic behavior of the National Aerospace Plane (NASP). Details of the code may be found in [1]. The simulation requires the solution of a large sparse symmetric positive definite system of order 16146. The solution method is a variant of Cholesky factorization followed by back substitution. The method is computationally efficient and very stable numerically for this class of matrices. However, identical simulations run on the Cray 2 and Cray Y-MP produce different results. The solution produced by the Cray Y-MP has two fewer correct decimal digits in the solution than the solution produced by the Cray 2. Output of identical runs on different computers collected by O. Storaasli is presented in the following table. *Max Displacement* is the largest component of the solution. *Norm of Residual* is the square root of the sum of the squares of the components of the residual vector using the computed solution. The results are listed in order of increasing residual.

Table 1

	Computer	Max Displacement	Norm of Residual
128-bit	Cray 2	0.44744034**1**	0.48E-18
64-bit	Convex 220	0.44744033**9**	0.24E-6
64-bit	IRIS	0.44744033**9**	0.34E-6
64-bit	IBM 3090	0.44744034**4**	0.12E-5
64-bit	Cray 2	0.4474403**03**	0.87E-5
64-bit	Cray Y-MP	0.44743**6106**	0.12E-3

The results show that the maximum displacement for most of the computer systems agree to eight decimal digits (rounded), the Cray 2 64 bit solution agrees to seven decimal digits, and the Y-MP agrees to five decimal digits. Given the similarity of the two Cray floating point arithmetic implementations, this result is unexpected. Some background is necessary to provide the framework for a discussion of the problem.

Floating Point Arithmetic Implementations

Floating point arithmetic implementations on current supercomputers vary widely among architectures. Supercomputer floating point number formats are nominally based on 64 bits. The various implementations may be described in terms of the number of bits assigned to the fractional part of the number (mantissa) and to the exponent, and the base (or radix) of the number system. The accuracy of computations in a given floating point implementation will be affected by mantissa length and the "cleanliness" of the arithmetic. *Clean* floating point arithmetic is defined by the property that all floating point operations performed in the arithmetic are accurate within one-half a unit in the last place [2][3]. IEEE 754 (along with IEEE 854) compliant floating point arithmetic requires the implementation in the arithmetic functional units of three extra low order digits, or in a binary machine, three extra bits. The first extra digit, the *guard digit*, prevents errors from being introduced in the result by the final left shift of the mantissa (postnormalization) in multiplication and subtraction. The second and third extra digits, the *round digit* and *sticky digits*, are required to implement correct symmetric rounding. The sticky digit may be implemented as a single extra bit regardless of base, hence is often referred to as a *sticky bit*. Common floating point arithmetic implementation deficiencies include imprecise or nonexistent rounding, lack of guard and/or round digits, and lack of a sticky bit. Combinations of these features can lead to errors of several or more bits for various floating point operations. For example, lack of a guard digit in subtraction can result in answers which are incorrect (most often) in the second to the last place. (It can be worse.)

Both the Cray-2 and the Cray Y-MP have a binary radix, 48 bits of precision in the mantissa, and lack a guard digit in subtraction. Neither machine performs correct symmetric rounding in multiplication. Addition and subtraction in the Y-MP is chopped, while in the Cray 2 it is *prerounded*. Prerounding eliminates the bias in error caused by chopping, but has significantly worse total error than true symmetric rounding. The unit roundoff for both machines is 2^{-48}, or about 10^{-15}.

Errors in the Solution of Linear Systems

Suppose we wish to solve an nxn positive definite system Ax=b. The error in the solution x is affected by two factors; the accuracy of the computer arithmetic used to solve the system, and the nearness of the linear system's matrix A to a singular matrix. The latter can be characterized by the *condition number* κ of the matrix. κ varies from a value of one (in the matrix 2-norm) for the best behaved matrices (orthogonal matrices), to infinity (in all matrix norms) for singular matrices.

Suppose the solution x of the nxn system Ax=b, A nonsingular, is desired. Gaussian elimination followed by back substitution produces a computed approximation $\hat{x}$ to x. An expression for the relative error e is

$$e = \frac{\|x - \hat{x}\|}{\|x\|}$$

Define the *unit roundoff* **u** by β^{1-t} where β is the base of the floating point arithmetic, and t is the number of mantissa bits. From classical error analysis, the relative error e resulting from the solution of a linear system by Gaussian elimination followed by back substitution can be as large as the product of the condition number and **u**:

(1)

$$e \leq \mathbf{u}\,\kappa$$

The residual r = b - A$\hat{x}$ is often used as measure of the accuracy of solution. The size of the residual can be related to the size of the of A and $\hat{x}$:

$$\|b-A\hat{x}\| \leq \mathbf{u}\|A\|\|\hat{x}\|$$

This equation says that Gaussian elimination produces an $\hat{x}$ for which the residual (though not necessarily the relative error) is small.

The IMSL scientific subroutine LFCQS was used to obtain an estimate of the condition number of greater than 10^{11} for the matrix from the NASP simulation, from which equation (1) suggests that there could be as few as four correct decimal digits in a result produced by the Cray systems. "Classical" error analysis usually leads to overly pessimistic bounds on the error in the solution. This is particularly true of Cholesky factorization. In practice, obtained accuracy is often much better than predicted[4][5][6], and for Cholesky factorization, tighter error bounds have been obtained[7]. The tighter error bounds, like the classical error bounds, have a multiplicative dependence on **u**. Thus, these error analyses fail to predict the difference in the accuracy obtained on the two Crays, which have the same **u**.

The Effect of Y-MP Floating Point Arithmetic on the Accuracy of the Computed Cholesky Factorization

Although the Cray 2 and Cray Y-MP share floating point storage formats, their respective floating point arithmetic implementations differ. In particular, the methods for computing floating point additions and subtractions are slightly different. It can be shown (see Appendix A) that the maximum error incurred in Cray 2 floating point subtraction is (usually) less than 1 in the last place. However, the Cray Y-MP floating point subtraction (usually) has errors less than +1 in the second to the last (47th) bit, or more than twice as large as the error of the Cray 2. This has important ramifications for the error behavior encountered when certain numerical algorithms on the Cray Y-MP, including the Cholesky factorization of symmetric positive definite matrices, are scaled up to larger problem sizes.

Cholesky factorization of a symmetric positive definite matrix A produces a lower triangular G with positive diagonal entries such that A = GG^T. (See, for instance, [8].) The original matrix A often has a positive main diagonal and mostly negative off-diagonal entries. During the Cholesky factorization process the diagonal entries of G are formed using subtraction operations, while the off diagonal entries of G are formed from mostly addition operations. The Cray Y-MP can produce results from subtract magnitude floating point operations that are as much as one in the second to the last place too large. The important point is that the error, if there is one, always makes the result a little larger. The Cray 2 subtract magnitude floating point operation, and clean arithmetic implementations such as the VAX, Convex, and IEEE 754, do not. As was pointed out by W. Kahan [9], making the main diagonal of the Cholesky factor G a little larger effectively changes the original matrix A, and induces a systematic error in the solution x of Ax=b.

It can be shown (see Appendix B) that with Cray Y-MP style floating point arithmetic, the *difference in relative error* Δe between solution components of model pde problems computed with Cray-2 style floating point and Y-MP style floating point can be as much as εn^2, where n is the size of the system, and ε is a small positive constant about the size of the unit roundoff **u**. This error is in addition to the error normally incurred in the factorization procedure. The error incurred in the solution of these model problems attributable to the factorization is bounded (usually quite loosely) by equation (1), and can be expected to scale O(n) as the order n of the matrix associated with the problem is increased. The scaling behavior O(n) of the error in the solution of the model problem as the problem size is increased is common to most computer systems; in particular, the Cray 2 exhibits this behavior. Numerical experiments conducted by N. McCown at NAS [10] found that the *observed difference in the relative error* ΔE between the Cray 2 and the Cray Y-MP in the solution of large banded positive definite matrix equations by Cholesky factorization fit a model equation of the form

$$\Delta E = cn^{\gamma}$$

where $c \cong \mathbf{u}$, and $0.8 \leq \gamma \leq 1.75$. This compares to the analytic prediction (found in Appendix B) of $\gamma \leq 2$ for the solution of the model problem using Cholesky factorization and Y-MP floating point arithmetic, and the prediction of $\gamma \leq 1$ for solutions obtained with most other floating point arithmetics.

The analytical model $\Delta e \leq \varepsilon n^2$ with $\varepsilon = 2^{-48}$ and n=16146 predicts a difference in relative error between the Cray 2 and Cray Y-MP of about 10^{-6}, which compares with the observed amount of

$$9.75E\text{-}6 = F((0.447440472 - 0.447436106),0.447440472)$$

for the NASP simulation.

The error imposed on Cholesky factorization by Y-MP floating point arithmetic increases significantly faster with increasing problem size than that produced by other floating point arithmetics. Since Cholesky is a highly efficient and well behaved numerical method for the solution of large positive definite systems, this is apparently an unsatisfactory situation.

Iterative Refinement

A technique for improving the accuracy of the approximate solution $\hat{x}$ obtained from a factorization such as Cholesky is *iterative refinement*. The incentive for using this algorithm lies in its relatively low computational cost compared with the computation of the original Cholesky factors. Suppose Ax=b has been solved by a factorization method and an approximate solution $\hat{x}$ has been produced. Compute:

$r = b - A\hat{x}$ (compute residual)

solve $GG^Te = r$ for e (using the previously computed factors)

$\hat{x} = \hat{x} + e$

Iterative refinement of an nxn system requires $O(n^2)$ flops, while the Cholesky factorization requires $O(n^3)$ flops. Thus, for large systems the computation of the factors dominates the total floating point operation count.

Traditionally, the computation of the residual has been performed in double precision, in which case the procedure is denoted *mixed precision iterative refinement* (MPIR). This stems from the observation that usually the factorization technique produces an $\hat{x}$ for which the residual is small, and thus susceptible to cancellation effects. Double precision computations on Cray computer systems are performed in software, and due to the nature of Cray floating point arithmetic, are performed at least three times slower [11] than they could be if implemented using clean hardware floating point arithmetic. Thus the use of MPIR to improve accuracy has not been as cost effective on Cray systems as on other systems with clean floating point arithmetic.

Recent work [12] suggests that a modification to the traditional MPIR approach can efficiently produce significant improvements in the accuracy of the approximate solution when the Cholesky factors are not as accurately computed. $\hat{x}$ Normally, the Cholesky algorithm produces a factorization that from the standpoint of classical error analysis is reasonably well behaved. The Y-MP produces a factorization that is not as accurate. In this case, *fixed precision iterative refinement* (FPIR) can be applied to the Y-MP factored matrix to improve the accuracy of the computed solution to nearly the maximum obtainable using clean floating point with the same mantissa size. FPIR is identical to MPIR, except that the residual is computed in single precision. FPIR was applied to the 16146 equation PVSOLVE problem and run on the NAS Cray 2 and Cray Y-MP. The results are summarized below. The 128-bit Cray 2 result (without refinement) and the 64 bit Y-MP MPIR result are presented for comparison purposes.

Table 2

	Computer	Max Displacement	Norm of Residual
128-bit	Cray 2	0.447440341	0.48E-18
64-bit	Cray 2	0.4474403**03**	0.87E-5
FPIR once		0.44744034**9**	0.39E-5
FPIR twice		0.44744034**4**	0.40E-5
64-bit	Cray Y-MP	0.44743**6106**	0.13E-3
FPIR once		0.4474402**70**	0.38E-5
FPIR twice		0.4474402**75**	0.35E-5
MPIR once		0.447440341	0.58E-6

From Table 2 it is clear that one application of FPIR improves the Y-MP solution to nearly the accuracy of the Cray 2 solution, and the Y-MP residual is smaller. Each application of FPIR adds about 4% of the solver cpu time to the total cpu time. FPIR does require increased memory requirements, however, as copies of both the initial matrix and the factor are required in memory for efficient execution of FPIR. One application of MPIR produces agreement with the Cray 2 128 bit result, at a cost of about 40% more cpu time over just the single precision solver alone.

Conclusion

The Cray 2 and Cray Y-MP floating point arithmetic implementations, though quite similar, have fundamental differences that cause observable differences in the output of NAS user codes. The Cray Y-MP implementation of floating point arithmetic has characteristics that cause significant degradation in the obtainable accuracy of the solution of positive definite systems by Cholesky factorization. This otherwise unsatisfactory situation is ameliorated by the existence of the computationally inexpensive FPIR algorithm that should provide nearly full precision results for linear systems solved using Y-MP arithmetic and Cholesky factorization.

Acknowledgments

The author wishes to acknowledge the comments and suggestions provided by John Barton and David Browning, of NAS RND; Horst Simon of NAS RNR; and Ron Sykora and Mike Ess of Cray Computer Corporation. The author also wishes to thank W. Kahan of the University of California at Berkeley for illuminating discussions on the interaction of the Cholesky factorization algorithm and floating point arithmetic.

Appendix A

The Subtract Magnitude Floating Point Operation on Cray 2 and Cray Y-MP

The following is a detailed analysis of the effect the specific hardware implementation of the subtract magnitude floating point operation of the Cray 2 and Cray Y-MP has on the accuracy of results.

Floating Point Numbers

The following analysis borrows heavily from [13]. Given integers r and p, define the set S(r,p) of floating point numbers to be *zero* and all numbers of the form

$$x = r^e m, \qquad \text{(A.1)}$$

where e is any integer (positive, negative, or zero) and m is a positive or negative fraction satisfying

$$r^{-1} \le |m| < 1$$

whose absolute value can be expressed in the base r using at most p digits.

That is,

$$|m| = r^{-p} M$$

where M is an integer in the range $r^{p-1} \le M < r^p$. In (A.1) the signed number m is called the *mantissa* of x and e is the *exponent* of x. Note that the exponent is unbounded. In actual floating point arithmetic implementations the exponent is bounded. Since the type of error associated with bounded exponents (overflow or underflow) is unimportant here, the distinction is neglected in the following analysis.

The floating point numbers may be viewed as real numbers, on which the standard arithmetic operations addition, subtraction, multiplication, and division may be performed. The result of these operations may not be in S(r,p), however. Since the result of floating point arithmetic must always be a floating point number, the floating point arithmetic operations must be defined. For our purposes, the definition of the *subtract magnitude* floating point operation on the Cray 2 and Cray Y-MP suffices. For these machines, r=2 and p=48. The subtract magnitude case arises if numbers having opposite signs are added or numbers having the same sign are subtracted.

The Cray 2 Subtract Magnitude Floating Point Operation

To find the floating point difference ab, where $a > 0 > b$, and $a \geq |b|$, put $a=r^e m$ and

$\oplus$

$b=r^f n$. Assume that a and b are normalized, i.e., the leading digits of m and n are not zero, unless a and b are zero. Then $b=r^e n'$, where $n'=r^{-(e-f)}n$ is obtained by shifting n to the right e-f places. We are assuming that there are enough digits in the register to hold all of the right-shifted nonzero digits of n'. Put n'' equal to the high order p digits of n'. (Bits shifted out of the p digit register are lost, as there is no guard digit.[14]) If the p+1st digit of n' is a 1, $n'''=n''+ 2^{-p}$, otherwise $n'''=n''$. (The truncated bits of the preshifted mantissa are rounded up.) Then

$$u' = m + n''' \leq m < 1.$$

If $u' = 0$, set a+b equal to a normalized zero. If u' is not zero, postnormalization may be required. Let k be the number of leading zeros in u'. Since $r^e u'=r^{e-k}(r^k u')$,

$$g=e-k \text{ and } u=r^k u' \text{ chopped.}$$

Then $ab=r^g u$

$\oplus$

Cray 2 Error Analysis

The *error* is defined to be the difference between the mantissa of the floating point result when computed in an infinite number of digits and the mantissa of the result when computed in the implemented floating point arithmetic.

The preshift requires that n shift right e-f places. If $e-f \geq 2$, $|n'''| \leq r^{-2}$, so

$$u'=m-|n'''| \geq r^{-1}-r^{-2} \geq r^{-2},$$

which implies that the number of leading zeros k is either 0 or 1. Therefore, if $k \geq 2$, then e-f must be 0 or 1. When $e-f = 0$ (no preshift), no error is introduced. When $e-f=1$, there are two possibilities: either a zero or a one in the p'th place was shifted right one place. If a zero was shifted right, $n'''=n''$, so no error occurs after postnormalization. If a one was right shifted, the answer obtained is one unit less than the exact answer in the p-k'th place after postnormalization.

Example 1

Assume r=2, p=5. Put
a = $.10000 \times 2^{0}$, b = $-.11001 \times 2^{-1}$
right shift b: n' = .01100**1**
n'' = .01100
round up n'''= .01101
subtract mantissas: $$\begin{array}{r} .10000 \\ \underline{-.01101} \\ .00011 \end{array}$$

so u' = .00011, k=3, u= .11000, but the exact mantissa, also produced by clean floating point arithmetic, is .111000 The error is .11100 - .11000 = .00100 = 2^{-p+k}.

If e-f≥2, then |n'''-n'| < r^{-p}. The maximum error obtained after postnormalization is less than plus or minus one unit in the last place.

Example 2

Assume r=2, p=3. Put
a = $.100 \times 2^{4}$, b = $-.111 \times 2^{2}$
right shift b: n' = .001**11**
n'' = .001
round up n'''= .010
subtract mantissas: $$\begin{array}{r} .100 \\ \underline{-.010} \\ .010 \end{array}$$

Then u' = .010, k = 1, u = .100, g = 4-1 = 3.

The error is .1001 - .100 = .0001 < 2^{-p}, and the result agrees with the result computed with clean floating point arithmetic.

Example 3

Assume r=2, p=3. Put
a = $.100 \times 2^{4}$, b = $-.101 \times 2^{0}$
right shift b: n' = .001**01**
n'' = .001
no round up n'''= .001
subtract mantissas: $$\begin{array}{r} .100 \\ \underline{-.001} \\ .010 \end{array}$$

So u' = .010, k = 1, u = .100, g = 4-1 = 3, and the error is
.1011 - .110 = -.0001 < -2^{-p}. This result agrees with the result computed with clean floating point arithmetic.

The Cray Y-MP Subtract Magnitude Floating Point Operation

To find the floating point difference ab, where a > 0 > b, and a ≥ | b | , put a=r^{e}m and

⊕

b=r^{f}n. Assume that a and b are normalized, i.e, the leading digits of m and n are not zero, unless a and b are zero. Then b=r^{e}n', where n'=$r^{-(e-f)}$n is obtained by shifting n to the right e-f places. Put n'' equal to the high order p digits of n'. (Bits shifted out of the p digit register are lost, as there is no guard digit.[15]) Unlike the Cray 2 subtract magnitude case, no roundup occurs, so n'''=n''. Then

$$u' = m + n''' \leq m < 1.$$

If u' = 0, set ab equal to a normalized zero. If u' is not zero, postnormalization may be required. Let k be the number of leading zeros in u'. Since $r^e u' = r^{e-k}(r^k u')$,

$$g=e-k \text{ and } u=r^k u' \text{ chopped.}$$

Then $ab=r^g u$

Y-MP Error Analysis

The preshift requires that n shift right e-f places. If e-f≥2, $|n'''|<r^{-2}$, so

$$u'=m-|n'''| \geq r^{-1}-r^{-2} \geq r^{-2},$$

which implies that the number of leading zeros k is either 0 or 1. Therefore, if k≥2, then e-f must be 0 or 1. When e-f=0 (no preshift), no error is introduced. When e-f=1, there are two possibilities: either a zero or a one was shifted right one place. If a zero was shifted right, no error occurs after postnormalization. If a one was right shifted, the answer obtained is one unit greater than the exact answer in the p-kth place, after postnormalization. This implies that the subtract magnitude result provided by the Cray Y-MP floating point can be as much as twice as large as the actual answer when severe cancellation occurs.

Example 4

Assume r=2, p=5. Put
a = .10000x2^1, b = -.11111x2^0
right shift b: n' = .01111$_1$
n'' = .01111
no round up n'''=.01111
subtract mantissas: .10000
$\underline{-.01111}$
.00001

So the computed difference is $r^{-p} = 2^{-5}$ but the exact result, also produced by clean arithmetic, is 2^{-6}. The Y-MP produces a result in this case that is twice as large as the exact result.

Example 5

Assume r=2, p=5. Put
a = .10000x2^0, b = -.11001x2^{-1}
right shift b: n' = .01100$_1$

n" = .01100

no round up n'"=.01100

subtract mantissas:

$$\begin{array}{r} .10000 \\ -.01100 \\ \hline .00100 \end{array}$$

So u' = .00100, k = 2, u = .10000, but the exact result, also produced by clean arithmetic, is .0111000 The error is .011100 - .10000 = .00010

If e-f $\geq$ 2, then n'"-n' < r^{-p}. Then u'-u < $r^{-(p-2)}$, that is, the error is less than one unit in the third to last place larger than the exact value.

Example 6

Assume r=2, p=3. Put

a = $.100x2^4$, b= $-.101x2^0$

right shift b: n' = .001**01**

n" = .001

no round up n'"= .001

subtract mantissas:

$$\begin{array}{r} .100 \\ -.001 \\ \hline .010 \end{array}$$

So u' = .010, k = 1, u = .100, g = 4-1 = 3, and the error is .1011 - .110 = -.0001 < -2^{-p}

Example 7

Assume r=2, p=3. Put

a = $.100x2^4$, b = $-.111x2^2$

right shift b: n' = .001**11**

n" = .001

no round up n'"=.001

subtract mantissas:

$$\begin{array}{r} .100 \\ -.001 \\ \hline .011 \end{array}$$

So u' = .011, k = 1, u = .110, g = 4-1 = 3, and the error is .1001 - .110 = -.011 < -2^{-p+2}

The result produced by clean arithmetic for this example is in error by less than 2^{-p}.

Appendix B

Relative Error and Perturbations to the Diagonal of the Matrix resulting from the Discretization of Laplace's Equation on a Square Domain.

This section derives the effect on the error of the solution of the discretized Laplacian on a square region due to adding a small positive constant ε to the diagonal entries of the matrix. Consider the square domain R: $0 < x < \pi$, $0 < y < \pi$, in which the numerical solution to the Dirichlet problem for the Laplace equation is to be obtained by employing the finite difference equation

(B.1) $F(SUP3(U_{i+1,j}-2U_{i,j}+U_{i-1,j}),h^2) + F(SUP3(U_{i,j+1}-2U_{i,j}+U_{i,j-1}),h^2) = 0$

With $h = \pi/N$, R is divided into N^2 square nets with $n=(N-1)^2$ interior points. Let A_h denote the square matrix formed from the five-point formula on the left-hand side of (B.1). Then the solution x of the system of finite difference equations (B.1) is given by $x = A_h^{-1}b$, where the right hand side vector b is derived by the application of boundary conditions on the boundary of the discretized problem domain. Let the n eigenvectors of $-A_h$ be denoted by $X_{pq}(ih,jh)$ with the corresponding positive eigenvalues υ_{pq}, p, q = 1, 2, ..., n-1.

From the defining relation

(B.2) $-A_hX_{pq} = u_{pq}X_{pq}$

we have

(B.3) $X_{pq} = \sin(pih)\sin(qih)$ p, q = 1, 2, ..., n-1

(B.4) $\upsilon_{pq} = F(1,h^2)(2\cos(ph) + 2\cos(qh) - 4)$ p, q = 1, 2, ..., n-1

The matrix $-A_h$ has diagonal elements equal to 4. If the diagonal elements are perturbed by the addition of a small positive constant ε, $\varepsilon << h$, the resulting matrix $-A_{h\varepsilon}$ has eigenvectors (B.3), and eigenvalues

(B.4)$\upsilon_{pq\varepsilon} = F(1,h^2)(2\cos(ph) + 2\cos(qh) - 4 - \varepsilon)$p, q = 1, 2, ..., n-1

Additionally, $-A_{h\varepsilon}$ is symmetric positive definite (as is $-A_h$). The n eigenvalues υ_{pq} and $\upsilon_{pq\varepsilon}$ may be labeled sequentially from 1 to n:

$\lambda_1 = \upsilon_{11}, \dots , \lambda_n = \upsilon_{N-1N-1}$

The eigenvectors (B.3) can be normalized to unit 2-norm length and ordered in such a way that the nxn matrix of eigenvectors U is unitary, and diagonalizes $A_{h\varepsilon}$, and the eigenvalues are ordered from largest to smallest:

. (B.5) $\quad U^T A_{h\varepsilon} U = \Lambda_\varepsilon, \qquad U^T U = I$

(B.6) $\quad \mathrm{diag}(\Lambda_\varepsilon) = [\lambda_{\varepsilon 1} \dots \lambda_{\varepsilon N}]^T, \qquad \lambda_{\varepsilon 1} \geq \lambda_{\varepsilon 2} \geq \dots \geq \lambda_{\varepsilon N}$

Then

(B.7) $\quad x = U\Lambda^{-1}U^T b, \qquad x_\varepsilon = U\Lambda_\varepsilon^{-1}U^T b$

Take norms to be 2-norms. Define the relative error Δe of the solution of the perturbed system by

$$\Delta e = F(||x-x_\varepsilon||,||x||)$$

then

$$\Delta e = F(||U\Lambda^{-1}U^T b-U\Lambda_\varepsilon^{-1}U^T b||,||x||)$$

$$= F(||U(\Lambda^{-1}-\Lambda_\varepsilon^{-1})U^T b||,||x||)$$

$$\leq F(||\Lambda^{-1}-\Lambda_\varepsilon^{-1}||||b||,||x||)$$

The 2-norm of the positive diagonal matrix $L = \Lambda^{-1}-\Lambda_\varepsilon^{-1}$ is the largest element of L. Put $C_{pq} = 2\cos(ph) + 2\cos(qh)$. The largest element of L is then

$$||L|| = A(\max,pq) \ |\upsilon_{pq}^{-1} - \upsilon_{pq\varepsilon}^{-1}|$$

$$= A(\max,pq) \ F(\varepsilon,(C_{pq} - 4)(C_{pq} - 4 - \varepsilon))$$

which occurs when p=q=1. Then

$$||L|| = F(\varepsilon,(4\cos(h) - 4)(4\cos(h) - 4-\varepsilon)) \cong \Phi(\varepsilon h^4,4) \cong \Phi(\varepsilon n^2,4)$$

So

$$\Delta e = F(||x-x_\varepsilon||,||x||) \leq F(\varepsilon n^2||b||,4||x||)$$

If

$$F(||b||,||x||) = O(1)$$

then

$$\Delta e \leq \varepsilon n^2$$

The relative error from the addition of a small positive amount ε to the diagonal entries of the matrix obtained from the discretized Laplacian can cause an error that increases as the square of the number of unknowns in the system. This may be contrasted to the expected amount of error from classical roundoff error analysis in eq. (1), from which it can be concluded that the relative error can be as much as the unit roundoff (**u**) times the matrix condition number. The condition number of the matrix of the discretized Laplacian is $O(h^{-2})=O(n)$ (see for instance [16]). This implies that the relative error in the solution of the system predicted by classical roundoff error analysis could be as much as proportional to n: $e \leq \mathbf{u}n$. Thus the amount of error incurred by adding the amount ε to the diagonal entries of the matrix rapidly dominates the total error.

References

[1] O. Storaasli, D.T. Nguyen, and T.K. Agarwall (1989), "Parallel-Vector Solution of Large Scale Structural Analysis Problems on Supercomputers," *AIAA Journal*, Vol. 28, No. 7. pp.1211-1216.

[2] D. Stevenson (1981). "A Proposed Standard for Binary Floating Point Arithmetic," *Computer 14 (March)*, 51-62.

[3] W.J. Cody, et al. (1984), "A Proposed Radix- and Word-length-independent Standard for Floating-point Arithmetic," *IEEE Micro*, August, 1984, pp. 86-100.

[4] W. Kahan (1966). "Numerical Linear Algebra," *Canadian Math. Bull.* 9, 757-801.

[5] T. F. Chan and D. E. Foulser (1988), "Effectively Well-Conditioned Linear Systems", *SIAM Journal of Scientific and Statistical Computing*, Vol. 9, No. 6.

[6] R. T. Haftka (1989), "Stiffness-Matrix Condition Number and Shape Sensitivity Errors," *AIAA Journal*, Vol. 28, No. 7. pp.1322-1324.

[7] J. Demmel (1989), "On Floating Point Errors in Cholesky," LAPACK Working Note 14.

[8] G. H. Golub, and C. F. Van Loan (1989). *Matrix Computations, Second Edition*, pp 126-127.

[9] W. Kahan, private communication.

[10] N. McCown, (1990) "A Numerical Study of Roundoff Error Differences Between the Cray-2 and Cray Y-MP", NAS RND Technical Report RND90-008.

[11] W. Kahan (1990). "How Cray's Arithmetic Hurts Scientific Computing", Cray User Group Meeting, Toronto, April, 1990

[12] M. Arioli, J. Demmel, and I. S. Duff. (1989). "Solving sparse linear systems with sparse backward error," *SIAM J. Matrix Anal. Appl.*, 10(2):165-190.

[13] P. H. Sterbenz (1974). *Floating Point Computation*, Prentice-Hall, Inc.

[14] Cray Research, Inc. (1989), *CRAY-2 Computer Systems Functional Description Manual, Revision D.*, p. 2-14.

[15] Cray Research, Inc. (1989), *CRAY Y-MP Computer Systems Functional Description Manual*, p. 2-16.

[16] W. F. Ames (1977) *Numerical Methods for Partial Differential Equations, Second Ed.*, Academic Press. p. 116.